ANDROID STUDIO – EXPERIENCE BASED PROJECT LEARNING

ChatConnect - A Real-Time Chat and Communication App

Submitted by

ELAVARASI.M	-	711022104016
LOGESHWARI.A	-	711022104029
KRISHNAVENI.B	-	711022104027
DHANUSH.M.P	-	711022104012

BACHELOR OF COMPUTER SCIENCE AND ENGINEERING

IN

FIFTH SEMESTER

COMPUTER SCIENCE AND ENGINEERING

INFO INSTITUTE OF ENGINEERING, COIMBATORE – 641107

NOVEMBER/DECEMBER - 2024

BONAFIDE CERTIFICATE

Certified that this project“ ChatConnect-A Real-Time Chat and Communication App“ is the Bonafide work of ELAVARASI.M (711022104016),LOGESHWARI.A(711022104029), KRISHNAVENI.B(711022104027),DHANUSH.M.P (711022104012) who carried out the project work under any supervision.

SIGNATURE

STAFF COORDINATOR

Mrs. A. SARANYA M.E.,

ASSISTANT PROFESSOR

DEPT. COMPUTER SCIENCE AND ENGINEERIN

INFO INSTITUTE OF ENGINEERING,

KOVILPALAYAM COIMBATORE - 641107

SIGNATURE

HEAD OF THE DEPARTMENT

Dr. G. SELVAVINAYAGAM Ph.D.,

HEAD OF THE DEPARTMENT

DEPT. COMPUTER SCIENCE AND ENGINEERING

INFO INSTITUTE OF ENGINEERING,

KOVILPALAYAM COIMBATORE - 641107

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We sincerely thank to Tamil Nadu Skill Development Corporation (TNSDC), Naan Mudhalvan" Platform and ANDROID STUDIO – EXPERIENCE BASED PROJECT LEARNING (EBPL) for encouragement towards our project work for providing necessary skill training.

We sincerely thank our Principal Dr. N. KOTTISWARAN, M.E., Ph.D., and Head of the Department Dr. G. SELVAVINAYAGAM, M.E., Ph.D., and also Staff Coordinator Mrs. A. SARANYA M.E for her encouragement towards our project works.

We also thank our project guide and our parents for the complete and whole hearted support, motivation guidance and help in making our project activities.

ABSTRACT

Simple Chatting Application

ChatConnect is a real-time messaging app developed with Android Studio, Kotlin, and Jetpack Compose. It integrates Firebase for backend services, providing features like user registration and real-time messaging. Data is persistently stored using Room Database, allowing for offline access. The app's UI is built with Jetpack Compose, leveraging its declarative approach for efficient state management and responsive layouts.

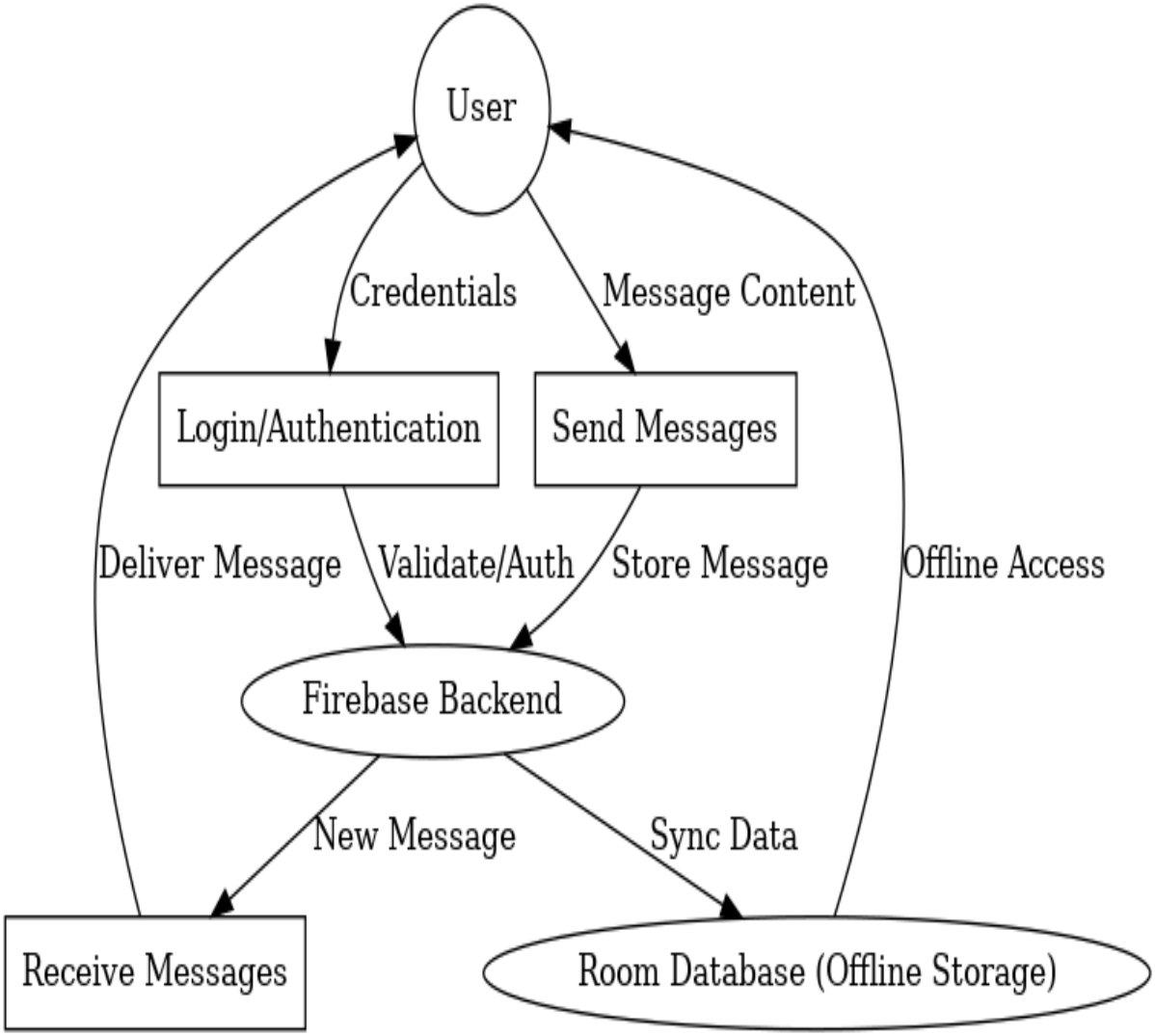
It supports seamless navigation and scalable design, ideal for modern Android applications. By working with Firebase and Room, developers can gain hands-on experience in backend integration and local storage. The project demonstrates best practices in building scalable, efficient apps. The app's architecture follows clean coding principles for maintainability. ChatConnect serves as a practical example of combining Firebase, Jetpack Compose, and local storage in Android app development.

INTDRODUCTION

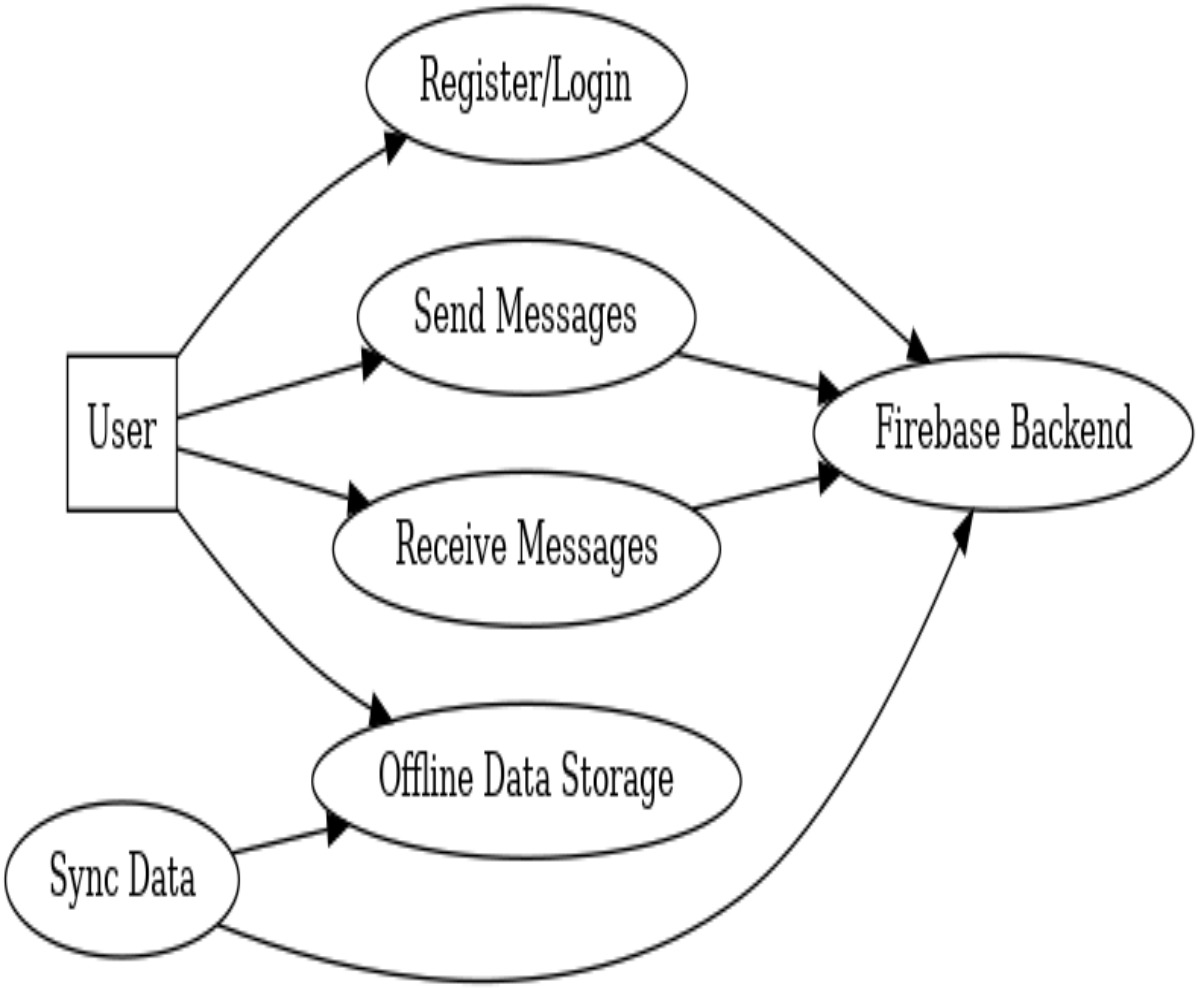
In today's digital age, communication apps have become essential tools for connecting people across the globe. These, messaging applications are among the most widely used, offering real-time communication and collaboration features. Chat Connect is a real-time chat and communication app designed to demonstrate the power and flexibility of Android development using modern tools and technologies.

Built with Android Studio and Jetpack Compose, Chat Connect allows users to engage in real-time conversations while leveraging advanced features such as Firebase integration for cloud-based services and Room Database for local data storage. This project showcases how developers can build scalable, efficient, and user-friendly messaging applications, making it an ideal resource for those looking to enhance their skills in Android app development. The app's architecture emphasizes the use of declarative UI, making it responsive, easy to maintain, and adaptable to various devices and screen sizes.

DATA FLOW DIAGRAM



USE CASE DIAGRAM



SOFTWARE REQUIREMENTS

1. Development: Windows, Android Studio, JDK 17, Kotlin

2. Android SDK: Min API 24, Target latest

3. UI: Jetpack Compose, Material Design 3

4. Database: Firebase ,Firestore

5. Backend: Firebase Auth, Realtime DB, FCM

6. Networking: Retrofit, Kotlin Serialization

7. Security: End-to-end encryption, HTTPS

8. Testing: JUnit, Espresso

9. Version Control: Git, GitHub Actions

10. Analytics: Firebase Crashlytics, Analytics

PROGRAM CODE

```
package com.project.flashchat.nav


import androidx.navigation.NavHostController
import
com.project.pradyotprakash.flashchat.nav.Destination.Home
import
com.project.pradyotprakash.flashchat.nav.Destination.Login
import
com.project.pradyotprakash.flashchat.nav.Destination.Register
object Destination {
    const val AuthenticationOption = "authenticationOption"
    const val Register = "register"
    const val Login = "login"
    const val Home = "home"
}
class Action(navController: NavHostController)
{
    val home: () -> Unit = {
        navController.navigate(Home) {
            popUpTo(Login) {
                inclusive = true
            }
            popUpTo(Register) {
                inclusive = true
```

```
            }
        }
    }
    val login: () -> Unit = { navController.navigate(Login) }
    val register: () -> Unit = { navController.navigate(Register) }
    val navigateBack: () -> Unit = { navController.popBackStack()
}
}
```

```
package com.project.pradyotprakash.flashchat.view

import androidx.compose.foundation.layout.fillMaxHeight
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material.*
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
```

```
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.VisualTransformation
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.project.pradyotprakash.flashchat.Constants

/**
 * Set of widgets/views which will be used throughout the application.
 * This is used to increase the code usability.
 */

@Composable
fun Title(title: String) {
    Text(
        text = title,
        fontSize = 30.sp,
        fontWeight = FontWeight.Bold,
        modifier = Modifier.fillMaxHeight(0.5f)
    )
}
```

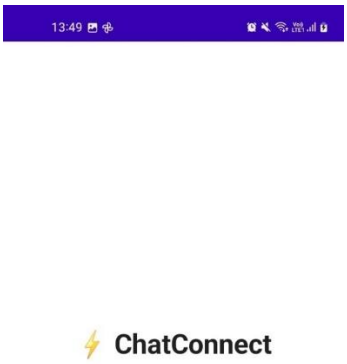
```
// Different set of buttons in this page
@Composable
fun Buttons(title: String, onClick: () -> Unit, backgroundColor:
Color) {
    Button(
        onClick = onClick,
        colors = ButtonDefaults.buttonColors(
            backgroundColor = backgroundColor,
            contentColor = Color.White
        ),
        modifier = Modifier.fillMaxWidth(),
        shape = RoundedCornerShape(0),
    ) {
        Text(
            text = title
        )
    }
}

@Composable
fun Appbar(title: String, action: () -> Unit) {
    TopAppBar(
        title = {
```

```
            Text(text = title)
        },
        navigationIcon = {
            IconButton(
                onClick = action
            ) {
                Icon(
                    imageVector = Icons.Filled.ArrowBack,
                    contentDescription = "Back button"
                )
            }
        }
    )
}

@Composable
fun TextFormField(value: String, onValueChange: (String) ->
Unit, label: String, keyboardType: KeyboardType,
visualTransformation: VisualTransformation) {
    OutlinedTextField(
        value = value,
        onValueChange = onValueChange,
        label = {
            Text(
```

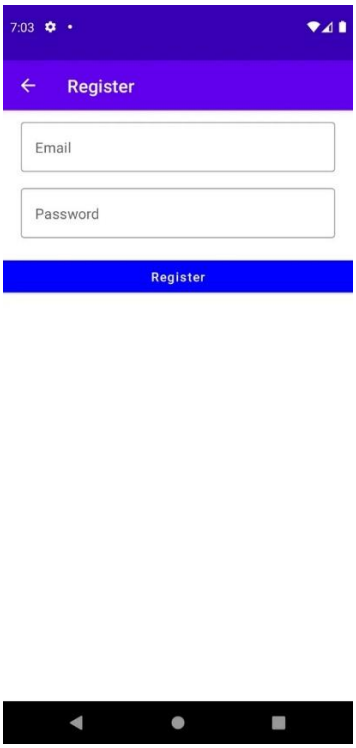
```
            label
        )
    },
    maxLines = 1,
    modifier = Modifier
        .padding(horizontal = 20.dp, vertical = 5.dp)
        .fillMaxWidth(),
    keyboardOptions = KeyboardOptions(
        keyboardType = keyboardType
    ),
    singleLine = true,
    visualTransformation = visualTransformation
  )
}

@Composable
fun SingleMessage(message: String, isCurrentUser: Boolean) {
  Card(
    shape = RoundedCornerShape(16.dp),
    backgroundColor = if (isCurrentUser)
MaterialTheme.colors.primary else Color.White
  ) {
    Text(
      text = message,
```

```
            textAlign =
            if (isCurrentUser)
                TextAlign.End
            else
                TextAlign.Start,
            modifier = Modifier.fillMaxWidth().padding(16.dp),
            color = if (!isCurrentUser) MaterialTheme.colors.primary
else Color.White
        )
    }
}
```

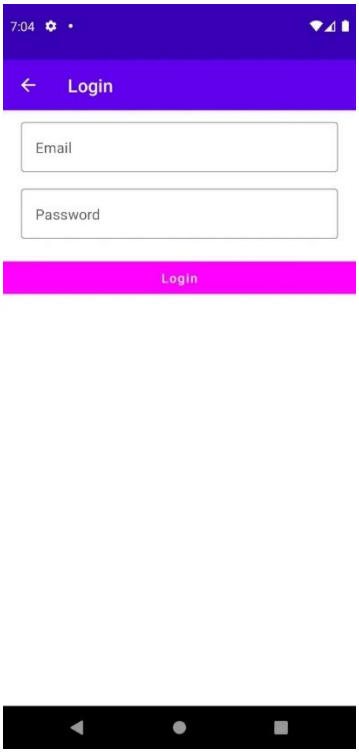
OUTPUT



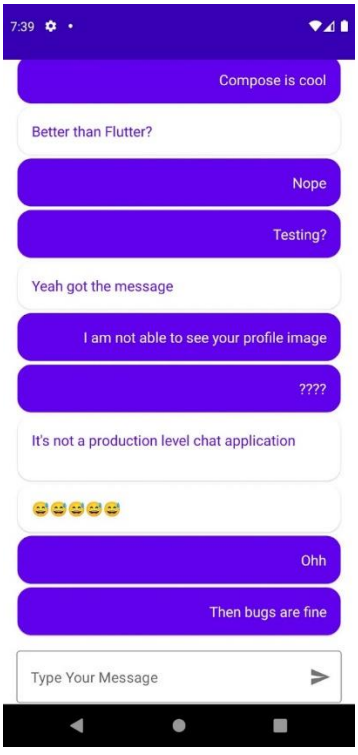
Home page



Register page



Login page



Chat Screen

Conclusion :

- ChatConnect is a powerful and innovative real-time chat application that combines modern development tools and robust functionalities to deliver an exceptional user experience.
- By utilizing Android Studio, Kotlin, Jetpack Compose, and Firebase, the app demonstrates effective implementation of real-time messaging, user authentication, and data persistence.
- Its intuitive design, responsive layouts, and efficient state management make it a reliable platform for communication. ChatConnect not only serves as a practical example of modern Android development but also offers scalability and security for personal and professional use. With planned future enhancements, it has the potential to become a comprehensive, multi-platform messaging solution catering to diverse user needs.

Future Enhancement:

- Future enhancements for ChatConnect include extending cross-platform compatibility to iOS and web applications, enabling a seamless multi-device experience. Security can be further strengthened with advanced end-to-end encryption, while real-time push notifications will improve user engagement.
- AI-powered features such as smart replies, sentiment analysis, and message scheduling can enhance usability. Voice assistant integration will provide hands-free convenience, and media optimization through automatic compression will ensure faster file transfers.
- Customizable themes and fonts will allow users to personalize their experience, and offline messaging capabilities will ensure messages are queued and sent upon reconnection.

- Group collaboration tools, including task management and polls, can make the app more versatile for professional use.
- Additionally, introducing premium features like expanded storage or advanced analytics can generate revenue and cater to power users.
- These enhancements will position ChatConnect as a cutting-edge, user-focused messaging platform.